ASR1K Frame Relay - Multilink (MLFR-FRF.16)

The ASR1K Frame Relay - Multilink (MLFR-FRF.16) feature is based on the Frame Relay Forum Multilink Frame Relay User-to-Network Interface/Network-to-Network Interface (UNI/NNI) Implementation Agreement (FRF.16.1) on Cisco Aggregation Services Routers. This feature provides a cost-effective way to increase the bandwidth for particular applications by enabling multiple serial links to be aggregated into a single bundle of bandwidth. Multilink Frame Relay (MFR) is supported on UNI in Frame Relay networks.

- Finding Feature Information, on page 1
- Prerequisites for ASR1K Frame Relay - Multilink (MLFR-FRF.16), on page 1
- Restrictions for ASR1K Frame Relay - Multilink (MLFR-FRF.16), on page 2
- Information About ASR1K Frame Relay - Multilink (MLFR-FRF.16), on page 3
- How to Enable ASR1K Frame Relay - Multilink (MLFR-FRF.16), on page 6
- Configuration Examples for ASR1K Frame Relay - Multilink (MLFR-FRF.16), on page 14
- Additional References, on page 15
- Feature Information for ASR1K Frame Relay - Multilink (MLFR-FRF.16), on page 16
- Glossary, on page 16

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the Feature Information for ASR1K Frame Relay - Multilink (MLFR-FRF.16), on page 16.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn . An account on Cisco.com is not required.

Prerequisites for ASR1K Frame Relay - Multilink (MLFR-FRF.16)

- MFR must be configured on the peer device.

Restrictions for ASR1K Frame Relay - Multilink (MLFR-FRF.16)

- Only the 2-octet Frame Relay format is supported.
- Only T1 and E1 speed members are supported in a bundle.
- All member links of a bundle must be of the same type.
- The following Shared Port Adapter (SPA) types are supported:
  - SPA-2XCT3/DS0
  - SPA-4XCT3/DS0
  - SPA-8XCHT1/E1
  - SPA-1XCHOC12/DS0
  - SPA-1XCHSTM1/OC3
- The following features are not supported with the ASR1K Frame Relay - Multilink (MLFR-FRF.16) feature:
  - 3- or 4-octet headers
  - Data-link connection identifier (DLCI) address mapping
  - Discard Eligibility (DE) bit manipulation
  - E1/T1 fractional links within the bundle
  - Frame Relay broadcast queue
  - Frame Relay backward explicit congestion notification (BECN) and forward explicit congestion notification (FECN) counting
  - Frame Relay Permanent Virtual Circuit (PVC) interface priority queuing (PIPQ) including DLCI prioritization
  - Frame Relay switching including NNI and FRF2.1
  - Frame Relay Traffic Policing (FRTP)
  - Frame Relay Traffic Shaping (FRTS)
  - FRF.16.1 Fragmentation
  - Generic Traffic Shaping (GTS)
  - Inverse Address Resolution Protocol (ARP)
  - PVC configuration over MFR bundle interface
  - Point-to-multipoint subinterfaces
  - Switched Virtual Circuits (SVC)
- An ISDN interface and any type of virtual interface cannot be a bundle link.
- The Multilink Frame Relay MIB (RFC 3020) is not supported.
- FRF.9 hardware compression over MFR is not supported.

Information About ASR1K Frame Relay - Multilink (MLFR-FRF.16)

Benefits of ASR1K Frame Relay - Multilink (MLFR-FRF.16)

Flexible Pool of Bandwidth

By combining multiple physical interfaces into a bundle, you can design a Frame Relay interface that has more bandwidth than is available from any single physical interface. For example, many new network applications require more bandwidth than is available on a T1 line. One option is to invest in a T3 line; however, T3 lines can be expensive and are not available in some locations. MFR provides a cost-effective solution to this problem by allowing multiple T1 lines to be aggregated into a single bundle of bandwidth.

Increased Service Resilience

When multiple physical interfaces are provisioned as a single bundle, they provide more service resilience than a single physical interface. If a link fails, the bundle continues to support the Frame Relay service by transmitting across the remaining bundle links.

Scalability

ASR1K supports up to 992 MFR bundles.

- MFR bundles—The following table shows the maximum number of MFR bundles supported on ASR1K based on the number of links in a bundle:

Table 1: Maximum MFR Bundles

Links per Bundle	Number of MFR Bundles
1	992
2	496
3	330
4	248

- Frame Relay DLCI—The number of Frame Relay DLCIs that can be configured on MFR subinterfaces equals the maximum number of MFR bundles on ASR1K. The maximum number of Frame Relay DLCIs that you can configure on MFR subinterfaces and in one MFR bundle is 992.
- MFR subinterface—Because only point-to-point interfaces are supported, the number of DLCIs supported is equal to the number of MFR subinterfaces. Therefore, the maximum number of MFR subinterfaces and the maximum number of MFR interfaces supported in one bundle is 992.
- Physical Links—The maximum number of physical links supported in a bundle is 10.

Link Integrity Protocol Control Messages

For link management, each end of a bundle link follows the MFR Link Integrity Protocol and exchanges link control messages with its peer (the other end of the bundle link). To bring up a bundle link, both ends of the link must complete an exchange of ADD_LINK and ADD_LINK_ACK messages. To maintain the link, both ends periodically exchange HELLO and HELLO_ACK messages. This exchange of hello messages and acknowledgments serve as a keepalive mechanism for the link. If a router is sending hello messages but not receiving acknowledgments, it will resend the hello message up to a configured maximum number of times. If the router exhausts the maximum number of retries, the bundle link line protocol is considered down (unoperational).

The bundle link interface's line protocol status is considered up (operational) when the peer device acknowledges that it will use the same link for the bundle. The line protocol remains up when the peer device acknowledges the hello messages from the local router.

The bundle interface's line status becomes up when at least one bundle link has its line protocol status up. The bundle interface's line status goes down when the last bundle link is no longer in the up state. This behavior complies with the class A bandwidth requirement defined in FRF.16.

The bundle interface's line protocol status is considered up when the Frame Relay data-link layer at the local router and peer device synchronize using the Local Management Interface (LMI), when LMI is enabled. The bundle line protocol remains up as long as the LMI keepalives are successful.

Variable Bandwidth Class Support

MFR FRF.16 variable bandwidth class support allows you to specify the criterion used to activate or deactivate a Frame Relay bundle.

Class A Single Link

The Frame Relay bundle is provisioned when one or more bundle links issue a BL_ACTIVATE message to indicate that an operational bandwidth is available. When this occurs, the bundle emulates a physical link by issuing a PH_ACTIVATE message to the data link layer.

When the operational bandwidth of a bundle link fails to meet operational requirements (for instance, if a bundle link is in rollback mode), the bundle link issues a BL_DEACTIVATE message. When all bundle links are down in a class A bundle, a PH_DEACTIVATE message is sent to the data link layer, indicating that the Frame Relay bundle cannot accept frames.

Note Activate and deactivate messages are implementation-oriented messages only. They are not visible in the output of the debug commands.

Class B All Links

The Frame Relay bundle is provisioned when all bundle links issue a BL_ACTIVATE message to indicate that an operational bandwidth is available. When this occurs, the bundle emulates a physical link by issuing a PH_ACTIVATE message to the data link layer.

When the operational bandwidth of a bundle link fails to meet operational requirements (for instance, if it is in loopback mode), the bundle link issues a BL_DEACTIVATE message. When any bundle link is down in

a class B bundle, a PH_DEACTIVATE message is sent to the data link layer, indicating that the Frame Relay bundle cannot accept frames.

Class C Threshold

A Frame Relay bundle is provisioned when a minimum number of links in the configured bundle issue a BL_ACTIVATE message, causing the bundle to emulate a physical link by issuing a PH_ACTIVATE message to the data link layer.

When the number of bundle links issuing a BL_ACTIVATE message falls below the configured threshold value, a PH_DEACTIVATE message is sent to the data link layer, indicating that the Frame Relay bundle cannot accept frames.

Load Balancing

MFR provides load balancing across bundle links within a bundle. If a bundle link that is chosen for transmission happens to be busy transmitting a long packet, the load-balancing mechanism can try another link, thus solving problems encountered when delay-sensitive packets have to wait.

ASR1K FRF.12 Support on MFR Interfaces

The ASR1K FRF.12 Support on MFR Interfaces feature enables the transport of realtime, delay-sensitive (voice) and nonrealtime, delay-insensitive (data) packets over the same, relatively slow-speed PVC.

During the transmission of packets, the larger, nonrealtime packets are fragmented into a sequence of smaller, mostly fixed-sized packets, also called fragments. The realtime packets are interleaved among the fragments. While receiving the packets, the nonrealtime fragments are reassembled and the resulting packets are forwarded along with the realtime packets. This approach minimizes the delay that can occur when nonrealtime and realtime traffic flow over the same PVC.

Benefits of ASR1K FRF.12

The ASR1K FRF.12 functionality prevents delay in Frame Relay networks by allowing edge routers to fragment large data packets before transmitting them across the network.

Limitations of ASR1K FRF.12

If a Frame Relay access device does not support FRF.12 fragmentation, the ASR1K FRF.12 Support on MFR Interfaces feature will not benefit the interface between the Frame Relay access device and the edge router. Fragmentation and reassembly occur on the interface between the edge router and the Frame Relay network.

If the Frame Relay access device is sending voice and unfragmented data on the same PVC, voice quality will suffer. The edge router will not reorder packets on PVCs.

Selecting a Fragment Size

You should set the fragment size based on the lowest port speed between routers. For example, for a hub-and-spoke Frame Relay topology, where the hub has a T1 speed and the remote routers have 64 kb/s port speeds, the fragmentation size must be set for 64 kb/s speed on both routers. Any other PVCs that share the same physical interface must use the same fragmentation size used by the voice PVC.

With pure end-to-end FRF.12 fragmentation, you should select a fragment size that is larger than the voice packet size.

The following table shows the recommended fragmentation sizes for a serialization delay of 10 ms:

Table 2: Recommended Fragment Size for 10 ms Serialization Delay

Lowest Link Speed in Path	Recommended Fragment Size
56 kb/s	70 bytes
64 kb/s	80 bytes
128 kb/s	160 bytes
256 kb/s	320 bytes
512 kb/s	640 bytes
768 kb/s	1000 bytes
1536 kb/s	1600 bytes

How to Enable ASR1K Frame Relay - Multilink (MLFR-FRF.16)

Configuring an MFR Bundle

Perform this task to configure an MFR bundle.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface mfr***interface-number*
4. **frame-relay multilink bandwidth-class** [**a** | **b** | **c** [*threshold*]]
5. **frame-relay intf-type** [**dce** | **dte**]
6. **frame-relay multilink bid** *name*
7. **exit**
8. **interface mfr***interface-number***.***subinterface-number* **point-to-point**
9. **ip address** *ip-address mask*
10. **frame-relay interface-dlci** *dlci*
11. **end**
12. **show frame-relay multilink**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface mfr***interface-number* **Example:** `Router(config)# interface mfr1`	Configures an MFR bundle interface and enters interface configuration mode.
Step 4	**frame-relay multilink bandwidth-class** [**a** | **b** | **c** [*threshold*]] **Example:** `Router(config-if)# frame-relay multilink bandwidth-class a`	(Optional) Specifies the bandwidth class criterion used to activate or deactivate a Frame Relay bundle. • Class A (single link)—The bundle will activate when any bundle link is up and deactivate when all bundle links are down (default). • Class B (all links)—The bundle will activate when all bundle links are up and deactivate when any bundle link is down. • Class C (threshold)—The bundle will activate when the minimum configured number of bundle links is up (the threshold) and deactivate when the minimum number of configured bundle links fails to meet the threshold. **Note** If no bandwidth class criterion is specified by using the **frame-relay multilink bandwidth-class** command, the Frame Relay bundle will default to class A (single link).
Step 5	**frame-relay intf-type** [**dce** | **dte**] **Example:** `Router(config-if)# frame-relay intf-type dce`	Configures a device to function as data communication equipment (DCE). • **dce**—(Optional) The router or access server functions as a switch connected to a router. • **dte**—(Optional) The router or access server is connected to a Frame Relay network. **Note** Only one end of a link should be configured as DCE. The other end will function as data terminal equipment (DTE), which is the default setting.
Step 6	**frame-relay multilink bid** *name* **Example:** `Router(config-if)# frame-relay multilink bid router1`	(Optional) Assigns a bundle identification name to an MFR bundle. • The bundle identification (BID) will not go into effect until the interface has gone from the “down” state to the “up” state. One way to bring the interface down and back up again is by using the **shutdown** and **no**

	Command or Action	Purpose
		shutdown commands in interface configuration mode (assuming that the physical state of the link is always up).
Step 7	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 8	**interface mfr***interface-number***.***subinterface-number* **point-to-point** **Example:** `Router(config)# interface mfr1.1 point-to-point`	Configures a point-to-point MFR subinterface and enters subinterface configuration mode.
Step 9	**ip address** *ip-address mask* **Example:** `Router(config-subif)# ip address 10.0.1.1 255.255.255.0`	Configures an IP address for the subinterface.
Step 10	**frame-relay interface-dlci** *dlci* **Example:** `Router(config-subif)# frame-relay interface-dlci 100`	Assigns a DLCI to a Frame Relay subinterface and enters Frame Relay DLCI configuration mode. • The DLCI range is from 16 to 1007.
Step 11	**end** **Example:** `Router(config-fr-dlci)# end`	Exits Frame Relay DLCI configuration mode and returns to privileged EXEC mode.
Step 12	**show frame-relay multilink** **Example:** `Router# show frame-relay multilink`	(Optional) Displays the current Frame Relay multilink configuration.

Configuring an MFR Bundle Link

Tip To minimize the latency that results from the arrival order of packets, Cisco recommends bundling physical links of the same line speed in one bundle.

Perform this task to configure an MFR bundle link.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface serial** *number*
4. **encapsulation frame-relay mfr***number* [*name*]

5. **frame-relay multilink lid** *name*
6. **frame-relay multilink hello** *seconds*
7. **frame-relay multilink ack** *seconds*
8. **frame-relay multilink retry** *number*
9. **end**
10. **show frame-relay multilink**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface serial** *number* **Example:** `Router(config)# interface serial 5/0`	Selects a physical interface and enters interface configuration mode.
Step 4	**encapsulation frame-relay mfr***number* [*name*] **Example:** `Router(config-if)# encapsulation frame-relay mfr1`	Creates an MFR bundle link and associates the link with a bundle.
Step 5	**frame-relay multilink lid** *name* **Example:** `Router(config-if)# frame-relay multilink lid first-link`	(Optional) Assigns a bundle link identification name to an MFR bundle link. • The bundle link identification (LID) is not functional until the interface has gone from the "down" state to the "up" state. One way to bring the interface down and back up again is by using the **shutdown** and **no shutdown** commands in interface configuration mode.
Step 6	**frame-relay multilink hello** *seconds* **Example:** `Router(config-if)# frame-relay multilink hello 9`	(Optional) Configures the interval in seconds after which a bundle link will send out hello messages. • The default value is 10 seconds.
Step 7	**frame-relay multilink ack** *seconds* **Example:** `Router(config-if)# frame-relay multilink ack 6`	(Optional) Configures the interval (in seconds) for which a bundle link will wait for a hello message acknowledgment before resending the hello message. • The default value is 4 seconds.

	Command or Action	Purpose
Step 8	**frame-relay multilink retry** *number* **Example:** `Router(config-if)# frame-relay multilink retry 3`	(Optional) Configures the maximum number of times a bundle link will resend a hello message while waiting for an acknowledgment. • The default value is 2 tries.
Step 9	**end** **Example:** `Router(config-if)# end`	Ends the configuration session and returns to privileged EXEC mode.
Step 10	**show frame-relay multilink** **Example:** `Router# show frame-relay multilink`	(Optional) Displays the current Frame Relay multilink configuration.

Configuring FRF.12 on an MFR Bundle Interface

Before you begin

You must create a class map and a policy map before enabling FRF.12 fragmentation of Frame Relay frames. For the class map, define a differentiated services code point (DSCP) value as the match criterion.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface mfr***interface-number*
4. **no ip address**
5. **frame-relay fragment** *fragment-size* **end-to-end**
6. **service-policy output** *policy-map-name*
7. **exit**
8. **interface mfr***interface-number***.***subinterface-number* **point-to-point**
9. **ip address** *ip-address mask*
10. **frame-relay interface-dlci** *dlci-value*
11. **end**
12. **show frame-relay fragment** [**interface** *interface* [*dlci*]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface mfr***interface-number* **Example:** `Router(config)# interface mfr1`	Configures an MFR bundle interface and enters interface configuration mode.
Step 4	**no ip address** **Example:** `Router(config-if)# no ip address`	Disables IP processing.
Step 5	**frame-relay fragment** *fragment-size* **end-to-end** **Example:** `Router(config-if)# frame-relay fragment 300 end-to-end`	Enables FRF.12 end-to-end fragmentation of Frame Relay frames. • The valid size range is from 16 to 1600. • To maintain low latency and low jitter for priority queue traffic, configure the fragment size to be greater than the largest high-priority frame that would be expected.
Step 6	**service-policy output** *policy-map-name* **Example:** `Router(config-if)# service-policy output pmap1`	Attaches a policy map to an output interface that is to be used as the service policy for that interface.
Step 7	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 8	**interface mfr***interface-number***.***subinterface-number* **point-to-point** **Example:** `Router(config)# interface mfr1.1 point-to-point`	Configures a point-to-point MFR subinterface and enters subinterface configuration mode.
Step 9	**ip address** *ip-address mask* **Example:** `Router(config-subif)# ip address 10.1.1.1 255.255.255.0`	Configures the IP address of the subinterface.
Step 10	**frame-relay interface-dlci** *dlci-value* **Example:** `Router(config-subif)# frame-relay interface-dlci 100`	Assigns a DLCI to the MFR subinterface and enters Frame Relay DLCI configuration mode. • The DLCI range is from 16 to 1007.

	Command or Action	Purpose
Step 11	**end** **Example:** `Router(config-fr-dlci)# end`	Exits Frame Relay DLCI configuration mode and returns to privileged EXEC mode.
Step 12	Required: **show frame-relay fragment** [**interface** *interface* [*dlci*]] **Example:** `Router# show frame-relay fragment`	(Optional) Displays statistics about Frame Relay fragmentation.

Monitoring and Maintaining MFR Bundles and Bundle Links

SUMMARY STEPS

1. **enable**
2. **debug frame-relay multilink** [**control** [**mfr***number* | **serial** *number*]]
3. **show frame-relay multilink** [**mfr***number* | **serial** *number*] [**detailed**]
4. **show interfaces mfr***number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug frame-relay multilink** [**control** [**mfr***number*	**serial** *number*]] **Example:** `Router# debug frame-relay multilink control mfr1`	(Optional) Displays debug messages for MFR bundles and bundle links.
Step 3	**show frame-relay multilink** [**mfr***number*	**serial** *number*] [**detailed**] **Example:** `Router# show frame-relay multilink mfr1 detailed`	(Optional) Displays configuration information and statistics about MFR bundles and bundle links.
Step 4	**show interfaces mfr***number* **Example:** `Router# show interfaces mfr1`	(Optional) Displays information and packet statistics for the bundle interface.

Examples

The following is sample output from the **show frame-relay multilink** command. Because a particular bundle or bundle link is not specified, information about all bundles and bundle links is displayed.

```
Router# show frame-relay multilink

Bundle: mfr1, State = down, class = A, fragmentation disabled
 BID = router1
 Bundle links:
  Serial3/1, HW state = Administratively down, link state = Down, LID = second-link
  Serial3/2, HW state = up, link state = Add_sent, LID = first-link
Bundle: mfr1, State = down, class = B, fragmentation disabled
 BID = router1
 Bundle links:
  Serial3/0, HW state = Administratively down, link state = Down, LID = third-link
  Serial3/1, HW state = Administratively down, link state = Down, LID = second-link
  Serial3/2, HW state = up, link state = Add_sent, LID = first-link
```

The following is sample output from the **show frame-relay multilink** command when a Frame Relay bundle is configured as bandwidth class C (threshold):

```
Router# show frame-relay multilink

Bundle: mfr2, State = down, class = C (threshold 100), fragmentation disabled
 BID = router2
 Bundle links:
  Serial3/1, HW state = Administratively down, link state = Down, LID = cisco2
  Serial3/0, HW state = Administratively down, link state = Down, LID = cisco1
```

The following is sample output from the **show frame-relay multilink** command when the **serial** *number* keyword and argument pair is specified. It displays information about the specified bundle link.

```
Router# show frame-relay multilink Serial 3/2

Bundle links:
  Serial3/2, HW state = up, link state = Add_sent, LID = first-link
  Bundle interface = mfr1,  BID = router1
```

The following is sample output from the **show frame-relay multilink** command when the **serial** *number* keyword and argument pair and the **detailed** option are specified. Detailed information about the specified bundle links is displayed.

```
Router# show frame-relay multilink Serial 3/2 detail

Bundle links:
  Serial3/2, HW state = up, link state = Add_sent, LID = first-link
  Bundle interface = mfr1,  BID = router1
    Cause code = none, Ack timer = 6, Hello timer = 9,
    Max retry count = 3, Current count = 0,
    Peer LID = , RTT = 0 ms
    Statistics:
    Add_link sent = 110, Add_link rcv'd = 0,
    Add_link ack sent = 0, Add_link ack rcv'd = 0,
    Add_link rej sent = 0, Add_link rej rcv'd = 0,
    Remove_link sent = 0, Remove_link rcv'd = 0,
    Remove_link_ack sent = 0, Remove_link_ack rcv'd = 0,
    Hello sent = 0, Hello rcv'd = 0,
    Hello_ack sent = 0, Hello_ack rcv'd = 0,
    outgoing pak dropped = 0, incoming pak dropped = 0
```

Configuration Examples for ASR1K Frame Relay - Multilink (MLFR-FRF.16)

Example: Configuring Multilink Frame Relay

The following example shows the configuration of bundle MFR1, where serial interfaces 3/0 and 3/2 are configured as bundle links:

```
interface MFR1
 no ip address
 frame-relay intf-type dce
 frame-relay multilink bid router1
!
interface MFR1.1 point-to-point
 ip address 10.0.0.1 255.255.255.0
 frame-relay interface-dlci 100
interface Serial3/0
 encapsulation frame-relay MFR1
 frame-relay multilink lid first-link
 frame-relay multilink hello 9
 frame-relay multilink retry 3
 frame-relay multilink ack 4
interface Serial3/2
 encapsulation frame-relay MFR1
 frame-relay multilink lid first-link
 frame-relay multilink hello 8
 frame-relay multilink ack 3
 frame-relay multilink retry 2
```

Example: Configuring Variable Bandwidth Class Support

The following example shows how to configure Frame Relay bundle MFR2 to use the class B (all links) criterion to get activated or deactivated:

```
interface MFR2
 frame-relay multilink bandwidth-class b
 frame-relay intf-type dce
 frame-relay multilink bid router2
 exit
interface MFR2.2 point-to-point
 ip address 10.1.1.10 255.255.255.0
 frame-relay interface-dlci 145
  end
```

Example: Configuring FRF.12 on an MFR Interface

The following example shows how to configure FRF.12 on an MFR interface:

```
class-map match-any tos_111
 match dscp cs7
policy-map voip
 class tos_111
  priority percent 100
interface mfr1
 frame-relay multilink bid 1
```

```
 frame-relay multilink bandwidth-class a
 frame-relay fragment 100 end-to-end
 service-policy output voip
interface mfr1.1 point-to-point
 ip address 70.1.1.1 255.255.255.0
 frame-relay interface-dlci 100
```

The following output shows the result of the above configuration:

```
Router# show frame-relay fragment

interface  dlci  frag-type   size  in-frag  out-frag  dropped-frag
mfr1.1     100   end-to-end  100   0        0         0
```

The size column displays the configured fragment size in bytes.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
WAN commands: complete command syntax, command mode, defaults, usage guidelines, and examples	Cisco IOS Wide-Area Networking Command Reference
Frame Relay configuration	Configuring Frame Relay

Standards and RFCs

Standard/RFC	Title
FRF.16.1	*Multilink Frame Relay UNI/NNI Implementation Agreement,* May 2002

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for ASR1K Frame Relay - Multilink (MLFR-FRF.16)

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 3: Feature Information for ASR1K Frame Relay - Multilink (MLFR-FRF.16)

Feature Name	Releases	Feature Information
ASR1K Frame Relay - Multilink (MLFR-FRF.16)	Cisco IOS XE Release 3.4S	The ASR1K Frame Relay - Multilink (MLFR-FRF.16) feature is based on the Frame Relay Forum Multilink Frame Relay UNI/NNI Implementation Agreement (FRF.16.1) on Aggregation Service Routers. The following commands were introduced or modified: **debug frame-relay multilink**, **encapsulation frame-relay mfr**, **frame-relay multilink ack**, **frame-relay multilink bandwidth-class**, **frame-relay multilink bid**, **frame-relay multilink hello**, **frame-relay multilink lid**, **frame-relay multilink retry**, **interface mfr**, **show frame-relay multilink**.
ASR1K FRF.12 Support on MFR Interfaces	Cisco IOS XE Release 3.5S	The following sections provide information about this feature: • ASR1K FRF.12 Support on MFR Interfaces • Configuring FRF.12 on an MFR Bundle Interface

Glossary

BID --Bundle identification. The BID is the name used to identify the bundle. The BID can be assigned, or the default can be used.

BL_ACTIVATE --A message that controls the addition of a bundle link to a Frame Relay bundle.

BL_DEACTIVATE --A message that controls the removal a bundle link from a Frame Relay bundle.

bundle --A logical grouping of one or more physical interfaces using the formats and procedures of multilink Frame Relay. A bundle emulates a physical interface to the Frame Relay data-link layer. The bundle is also referred to as the *MFR interface* .

bundle link --An individual physical interface that is a member of a bundle.

DLCI --data-link connection identifier. A value that identifies a permanent virtual circuit (PVC) in a Frame Relay network.

HELLO message --A message that notifies a peer endpoint that the local endpoint is in the operational state (up).

HELLO_ACK --A message that notifies a peer endpoint that a hello message has been received.

LID --link identification. The LID is the name used to identify a bundle link. The LID can be assigned, or the default can be used.

LMI --Local Management Interface. A set of enhancements to the basic Frame Relay specification. LMI includes support for a keepalive mechanism, which verifies that data is flowing; a multicast mechanism, which provides the network server with its local DLCI and the multicast DLCI; global addressing, which gives DLCIs global rather than local significance in Frame Relay networks; and a status mechanism, which provides an ongoing status report on the DLCIs known to the switch.

NNI --Network-to-Network Interface. The interface between two Frame Relay devices that are both located in a private network or both located in a public network.

PH_ACTIVATE --A message that indicates that the Frame Relay bundle is up.

PH_DEACTIVATE --A message that indicates that the Frame Relay bundle is down.

UNI --User-to-Network Interface. The interface between a Frame Relay device in a public network and a Frame Relay device in a private network.